UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/628,054	07/25/2003	Vinod K. Grover	3382-65598-01	4135

26119 7590 05/12/2009

KLARQUIST SPARKMAN LLP
121 S.W. SALMON STREET
SUITE 1600
PORTLAND, OR 97204

EXAMINER	
TECKLU, ISAAC TUKU	
ART UNIT	PAPER NUMBER
2192	
MAIL DATE	DELIVERY MODE
05/12/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/628,054	GROVER ET AL.	
	Examiner	Art Unit	
	ISAAC T. TECKLU	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *12 February 2009*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-15,17-23,26-32,34-37,39-41 and 43* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-15, 17-23, 26-32, 34-37, 39-41 and 43* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the amendment filed on 02/12/2009.

2. Claims 16, 24-25, 33, 38 and 42 have been cancelled.

3. Claims 1-15, 17-23, 26-32, 34-37, 39-41 and 43 have been examined.

Response to Arguments

4. Applicant's arguments with respect to claims 1-23, 26-32, 34-37, 39-41 and 43 have been considered but are moot in view of the new ground(s) of rejection. See Gordon et al. (US 6,560,774 B1) and Burger et al. (US 2004/0268327 A1), arts made of record.

Claim Rejections - 35 USC § 102

5. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

Claims 1-4, 5-12, 15, 17-23, 26-27, 30-32, 34-37, 39-41 and 43 are rejected under 35 U.S.C. 102(a) as being anticipated by Gordon et al. (US 6,560,774 B1 – art made of record).

Per claim 1 (Currently Amended), Gordon discloses one or more computer-readable media with computer-executable instructions for implementing a software development architecture (e.g. see at least FIG. 1, 31 and related text) comprising:

a software development scenario-independent intermediate representation format (e.g. see at least FIG. 2, IL CODE and related text);

one or more exception handling models operable to support a plurality of programming language specific exception handling models for a plurality of different source languages; (col. 12:15-25 "... verifier must ... determine conservative type ... exception handler ..." col. 25:40-50 "... Handling exceptions, including cross-language exceptions ..." and e.g. see at least FIG. 2, VB, VC++, OTHER and related text);

a type system operable to represent the type representations of a plurality of source languages (e.g. FIG. 10, Type and Purpose and related text); and

a code generator operable to generate code targeted for a plurality of execution architectures (e.g. see at least FIG. 2, EXECUTION ENGINE 200 and related text);

wherein the code generator constructs one or more software development components of software development tools using the software development scenario-independent intermediate representation format (col. 12:15-25 "... verifier must ... determine conservative type ... exception handler ..." col. 25:40-50 "... Handling exceptions, including cross-language exceptions ..." and e.g. see at least FIG. 2, VB, VC++, OTHER and related text), the one or more exception handling models operable to support the plurality of programming language specific exception handling models for the plurality of different source languages, and the type system operable to represent the plurality of different source languages (see at least col.12:5-15 "... exception handler...", col.12:20-25, col.24:60-65, e.g. FIG. 3, FIG. 23, Linker and related text).

Per claim 4, Gordon discloses the one or more computer-readable media of claim 1 wherein the software development architecture is operable to produce a software development tool modifiable by combining a modification component with the software development architecture (col. 26:1-25 "… class loader loads the implementation of a class …").

Per claim 5, Gordon discloses the one or more computer-readable media of claim 1 wherein the software development architecture is operable to produce a software development tool by dynamically linking a binary version of the software development architecture to a modification component (e.g. see at least FIG. 23, Linker and related text).

Per claim 6, Gordon discloses the one or more computer-readable media of claim 1 wherein the intermediate representation format is extensible at runtime of a software tool employing the intermediate representation format (e.g. see at least FIG. 23, IL & Metadata and related text).

Per claim 7, Gordon discloses the one or more computer-readable media of claim 1 wherein the architecture is combinable with one or more software development components (col. 26:1-25 "… EE's class loader loads the implementation of a class …").

Per claim 8, Gordon discloses the one or more computer-readable media of claim 7 wherein the one or more software development components comprise data describing a target software development tool (col. 12:15-25 "… verifier must … determine conservative type …

exception handler …" col. 25:40-50 "… Handling exceptions, including cross-language exceptions …" and e.g. see at least FIG. 2, VB, VC++, OTHER and related text).

Per claim 9, Gordon discloses the one or more computer-readable media of claim 7 wherein the one or more software development components provides target execution architecture data to the code generator (e.g. see at least FIG. 1, 35, 21 and related text).

Per claim 10, Gordon discloses the one or more computer-readable media of claim 7 wherein the one or more software development components provide one or more type-checking rules to the type system (e.g. see at least FIG. 23, Verifier and related text).

Per claim 11, Gordon discloses the one or more computer-readable media of claim 7 wherein the one or more software development components provide a set of class extension declarations to the architecture (e.g. see at least FIG. 2, 200 and related text).

Per claim 12, Gordon discloses the one or more computer-readable media of claim 7 wherein the combined one or more software development components and architecture produce a target software development tool (col. 25:25-40 "… Execution Engine … Code management … software memory isolation … ", and optimization (col. 25:30-55 "… insertion and execution of security checks …hardware specific optimization ….").

Per claim 15, Gordon discloses a method of creating a target software development tool, the method comprising:

receiving at least one computer-readable specification specifying functionality specific to one or more software development scenarios (col. 12:15-25 "… verifier must … determine conservative type … exception handler …" col. 25:40-50 "… Handling exceptions, including cross-language exceptions …" and e.g. see at least FIG. 2, VB, VC++, OTHER and related text);

creating at least one software development component from the at least one specification; and integrating the at least one software development component into a software development scenario-independent framework (e.g. FIG. 2, 200 and related text).

wherein the computer-readable specification comprises functionality for processing an intermediate representation format capable of representing a plurality of different programming languages; and (e.g. see at least FIG. 2, IL CODE and related text)

wherein the intermediate representation format comprises one or more exception handling models capable of supporting a plurality of programming language-specific exception handling models for the plurality of different programming languages (col. 12:15-25 "… verifier must … determine conservative type … exception handler …" col. 25:40-50 "… Handling exceptions, including cross-language exceptions …" and e.g. see at least FIG. 2, VB, VC++, OTHER and related text).

Per claim 17, Gordon discloses the method of claim 15 wherein software development components created from a plurality of computer-readable specifications for a plurality of

respective software development scenarios are integrated into the framework (e.g. see at least FIG. 23, JIT compilers and related text).

Per claim 18, Gordon discloses the method of claim 17 wherein the plurality of computer-readable specifications specify functionality for the following respective software development scenarios: target execution architecture; input language or input binary format (e.g. FIG. 2, 200 and related text; and compilation type (e.g. FIG. 23, JIT compilers and related text).

Per claim 19, Gordon discloses the method of claim 15 wherein the computer-readable specification specifies functionality for target execution architecture of the software development tool (e.g. FIG. 23, JIT compilers and related text).

Per claim 20, Gordon discloses the method of claim 15 wherein the computer-readable specification specifies functionality for accommodating an input language for the software development tool (e.g. see at least FIG. 2, VB, VC++, OTHER and related text).

Per claim 21, Gordon discloses the method of claim 15 wherein the computer-readable specification specifies functionality for accommodating a binary input for the software development tool (e.g. see at least FIG. 2, VB, VC++, OTHER and related text).

Per claim 22, Gordon discloses the method of claim 15 wherein the computer-readable specification comprises one or more rule sets for type-checking one or more languages (e.g. see at least FIG. 23, Verifier and related text).

Per claim 23, Gordon discloses the method of claim 15 wherein the computer-readable specification comprises a set of class extension declarations specific to one or more of the software development scenarios (col. 26:1-25 "… EE's class loader loads the implementation of a class …").

Per claim 26 (Currently Amended), Gordon discloses the method of claim 15 wherein the intermediate representation comprises type representations capable of representing the type representations of the plurality of different programming languages (e.g. see at least FIG. 2, VB, VC++, OTHER and related text).

Per claim 27, Gordon discloses the method of claim 15 further comprising: integrating custom code specific to one of the software development scenarios (e.g. see at least FIG. 2, VB, VC++, OTHER and related text).

Per claim 30, Gordon discloses one or more computer-readable media containing one or more computer-executable instructions for performing the method of claim 15 (col. 26:1-25 "… EE's class loader loads the implementation of a class …").

Per claim 31 (Currently Amended), Gordon discloses a method of creating a target software development tool from a common framework, the method comprising: configuring the common framework based on one or more characteristics of the target software development tool;

integrating software development components data comprising one or more characteristics of the target software development tool into the common framework (col. 26:1-25 "… EE's class loader loads the implementation of a class …"); and

creating the target software development tool from the integrated common framework (e.g. FIG. 23, JIT compilers and related text).

wherein the one or more characteristics comprises an input language chosen from a plurality of different programming languages supported by the common framework for the target software development tool and (e.g. see at least FIG. 2, VB, VC++, OTHER and related text);

wherein the common framework comprises exception handling models capable of supporting a plurality of programming language-specific exception handling models for the plurality of different programming languages (col. 12:15-25 "… verifier must … determine conservative type … exception handler …" col. 25:40-50 "… Handling exceptions, including cross-language exceptions …" and e.g. see at least FIG. 2, VB, VC++, OTHER and related text).

Per claim 32 (Currently Amended), Gordon discloses the method of claim 31 wherein the one or more characteristics can further comprise the amount of memory necessary for the target

software development tool to execute on a target architecture, the speed at which the target software development tool will execute on a target architecture, a input binary format for the target software development tool, or the target architecture for the target software development tool to execute on a target architecture (col. 12:15-25 "… verifier must … determine conservative type … exception handler …" col. 25:40-50 "… Handling exceptions, including cross-language exceptions …" and e.g. see at least FIG. 2, VB, VC++, OTHER and related text).

Per claim 34 (Currently Amended), Gordon discloses a method of producing inter-compatible software development tools, the method comprising:

creating a first software development tool by integrating software development components into a software development architecture that is operable to support a plurality of different programming languages; (e.g. FIG. 23, JIT compilers and related text); and

creating a second software development tool based on the first software development tool, wherein the second software development tool dynamically links to a binary version of the software development architecture (e.g. FIG. 23, Linker and related text).

wherein the software development architecture comprises functionality for exception handling models operable to support programming-language specific exception handling models for the plurality of different programming languages, and the software development architecture is used by both the first and second software development tools (col. 12:15-25 "… verifier must … determine conservative type … exception handler …" col. 25:40-50 "…

Handling exceptions, including cross-language exceptions …" and e.g. see at least FIG. 2, VB, VC++, OTHER and related text).

Per claim 35, Gordon discloses the method of claim 34 wherein the binary version of the software development architecture contains classes that are extensible through a set of declarations (e.g. FIG. 2, 200 and related text).

Per claim 36, Gordon discloses the method of claim 34 wherein the software development architecture comprises functionality for an intermediate representation format used by both the first and second software development tools (e.g. FIG. 2, IL code and related text).

Per claim 37, Gordon discloses the method of claim 34 wherein the software development architecture comprises functionality for a type system used by both the first and second software development tools (col. 12:15-25 "… verifier must … determine conservative type … exception handler …" col. 25:40-50 "… Handling exceptions, including cross-language exceptions …" and e.g. see at least FIG. 2, VB, VC++, OTHER and related text).

Per claim 39, Gordon discloses a method of modifying a software development tool, the software development tool having been created using a software development architecture that is operable to support a plurality of different programming languages and comprising one or more software development components, the method comprising:

dynamically linking a software development component not present in the software development architecture to a binary version of the software development architecture (e.g. FIG. 23, Linker and related text); and

creating a modified software development tool from the dynamically linked binary version and the software development component (e.g. FIG. 23, Linker and related text).

wherein the binary version of the software development architecture comprises functionality for exception handling models operable to support a plurality of programming language specific exception handling models for the plurality of different programming languages used by the modified software development tool (col. 12:15-25 "... verifier must ... determine conservative type ... exception handler ..." col. 25:40-50 "... Handling exceptions, including cross-language exceptions ..." and e.g. see at least FIG. 2, VB, VC++, OTHER and related text).

Per claim 40, Gordon discloses the method of claim 39 wherein the binary version of the software development architecture comprises classes that are extensible through a set of declarations (e.g. FIG. 2, 200 and related text).

Per claim 41, Gordon discloses the method of claim 39 wherein the binary version of the software development architecture comprises functionality for a type system used by the modified software development tool (e.g. FIG. 2, Executable code and related text).

Per claim 43 (Currently Amended), Gordon discloses a method of creating a software development tool, the method comprising:

receiving at least one computer-executable file (and e.g. see at least FIG. 2, VB, VC++, OTHER and related text) comprising:

an intermediate representation capable of representing a plurality of different programming languages and computer executable images (e.g. FIG. 2, IL CODE and related text);

one or more exception handling models capable of supporting a plurality of programming language specific exception handling models for the plurality of different programming languages; (col. 12:15-25 "... verifier must ... determine conservative type ... exception handler ..." col. 25:40-50 "... Handling exceptions, including cross-language exceptions ..." and e.g. see at least FIG. 2, VB, VC++, OTHER and related text);

a type system capable of representing the type representations of a plurality of source languages (e.g. FIG. 10, Type and Purpose and related text); and a code generator capable of generating code targeted for a plurality of execution architectures (e.g. FIG. 2, 200 and related text);

linking a software development component to the at least one computer-executable file using at least one class extension declaration (e.g. FIG. 23, Linker and related text); and

creating the software development tool via the linked software development component and computer-executable file (e.g. FIG. 2, 200 and related text).

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 2-3, 13-14 and 28-29 are rejected under 35 U.S.C. 103(a) as being unpatentable over Gordon (US 6,560,774 B1) in view of Burger et al. (US 2004/0268327 A1).

Per claims 2, 13-14 and 28-29 Gordon substantially disclosed the invention as claimed. However, Gordon was silent regarding the architecture to be scalable to produce target software development tools ranging from lightweight just-in-time (JIT) compilers. Nevertheless as evidenced by the teaching of (paragraph [0003] "… development tool such as compilers, debuggers and optimizers...", [0062]-[0063] "… software development tool can be retargeted between different architecture…JIT compilers…" and e.g. FIG. 4, 442A-442N) . Therefore it would have been obvious to one skilled in the art at the time the invention was made to have the software development tools retarget able or scalable to automatically generate software development tools for maintaining uniformity across architectures even within the same source base in order to correct any defects detected in the code by automatically propagating to other tools as once suggested by Burger (paragraph [0007] - [0008]).

Per claim 3, Gordon substantially disclosed the invention as claimed. However, Gordon was silent regarding the architecture can be configured to produce a target software

development tool with varying ranges of memory footprint, compilation speed. Nevertheless as evidenced by the teaching of (paragraph [0003] "… development tool such as compilers, debuggers and optimizers...", [0062]-[0063] "… software development tool can be retargeted between different architecture…JIT compilers…", paragraph [0131] "… memory operands specify any type…" and e.g. FIG. 4, 442A-442N) . Therefore it would have been obvious to one skilled in the art at the time the invention was made to have the software development tools retarget able or scalable to automatically generate software development tools for maintaining uniformity across architectures even within the same source base in order to correct any defects detected in the code by automatically propagating to other tools as once suggested by Burger (paragraph [0131])

Conclusion

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to ISAAC T. TECKLU whose telephone number is (571) 272-7957. The examiner can normally be reached on M-TH 9:300A - 8:00P.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Isaac T Tecklu/
Examiner, Art Unit 2192

/Tuan Q. Dam/
Supervisory Patent Examiner, Art Unit 2192